TILE LATCHING SYSTEMS AND PROBABILISTIC CONJUNCTIVE GRAMMAR

A SYNOPSIS

Submitted by

KANCHAN DEVI K

(Reg. No. 201613127)

in partial fulfillment for award of the degree

of

DOCTOR OF PHILOSOPHY

DEPARTMENT OF MATHEMATICS

SCHOOL OF ADVANCED SCIENCES

KALASALINGAM ACADEMY OF RESEARCH AND EDUCATION

(DEEMED TO BE UNIVERSITY)

ANAND NAGAR, KRISHNANKOIL - 626 126

TAMILNADU, INDIA

JUNE 2024

TILE LATCHING SYSTEMS AND PROBABILISTIC CONJUNCTIVE GRAMMAR

This thesis embodies the work done by K. Kanchan Devi under the guidance of Professor S. Arumugam.

The thesis consists of seven chapters.

1. Introduction
2. Triangular Tile Latching System
3. Single-annular tiling patterns
4. Probabilistic conjunctive grammar
5. Assignment of Probabilities to the Production Rules of a Conjunctive Grammar
6. Password Cracking Algorithm using Probabilistic Conjunctive Grammar
7. Conclusion

For basic graph theoretic terminology we refer to Chartrand and Lesniak [1]. For basic definitions in formal language we refer to [2, 3] and [4].

By a graph $G = (V, E)$, we mean a finite undirected graph with neither loops nor multiple edges. The order $|V|$ and size $|E|$ of

G are denoted by n and m respectively. A path of order n, a cycle of order n and a complete graph of order n are denoted by P_n, C_n and K_n respectively.

A connected acyclic graph is called a *tree*. A graph having exactly one cycle is called a unicyclic graph. A disconnected graph in which each component is a tree is called a *forest*.

A graph $G = (V, E)$ is called a *planar graph* if it can be drawn in a plane in such a way that no two edges in G intersect in a point other than a vertex of G, where each edge of G is a simple arc or a Jordan arc. Such a drawing of G is called a *plane graph*. A plane graph divides the rest of the plane into regions, which are called the *faces* of G. Exactly one face of G is unbounded and is called the *exterior face* of G. The remaining faces are called *interior faces*. The number of faces in a plane graph is denoted by r.

The following theorem is known as Euler's polyhedron formula.

Theorem 1. [1] *Let G be a connected plane graph with n vertices, m edges and r faces. Then $n - m + r = 2$.*

Let G be a plane graph. Its geometric dual G^* is constructed by placing one vertex in each region of G and if two regions have a common edge x, then the corresponding vertices are joined by an edge x^* such that x^* crosses only x.

We now proceed to give some basic definitions and theorems in formal language theory.

An alphabet is a finite nonempty set Σ whose elements are called *symbols*. A sequence of symbols from Σ is called a *string*. The empty string, denoted by ϵ, is the string that contains no symbols. If x and y are strings over Σ, then the string obtained by concatenating x and y is denoted by xy.

Trivially $x\epsilon = \epsilon x = x$ for any string x.

Let Σ^* be the set of all strings over the alphabet Σ. Any subset $L \subseteq \Sigma^*$ is called a *language over* Σ. The language L is called *finite* or *infinite* according as $|L|$ is finite or infinite.

Since a language is a subset of Σ^*, it follows that if L_1 and L_2 are languages over Σ, then $L_1 \cup L_2, L_1 \cap L_2$ and $L_1 - L_2$ are languages over Σ. Also $L_1L_2 = \{w_1w_2 : w_1 \in L_1 \text{ and } w_2 \in L_2\}$ is called the *concatenation* of L_1 and L_2.

Let $A, B \in \Sigma^*$. Then the ordered pair (A, B) is called a *rewriting rule* or *production rule* and is denoted by $A \to B$. Let P be a finite set of production rules. If $A_1 \to B_1$ is a production rule and if A and B are two strings of the form $A = A'A_1A''$ and $B = A'B_1A''$, then we say that A directly generates B. In other words B can be obtained from A by applying the production rule $A_1 \to B_1$. In this case we write $A \Rightarrow B$.

Let P be a finite set of production rules. We say that a string A generates a string B if there exists a finite sequence of strings $D = (A_0, A_1, A_2, \ldots, A_k)$ in Σ^* such that $A_0 = A, A_k = B$ and A_i generates directly A_{i+1} for $0 \leq i \leq k-1$. If A generates B, then we write $A \overset{*}{\Rightarrow} B$. The sequence D is referred to as a derivation of B from A and k is called the length of the derivation.

A generative grammar is a quadruple $G = (\Sigma, N, P, S)$ satisfying the following conditions.

(i) Σ and N disjoint nonempty sets, called the set of terminals and the set of nonteriminals respectively.

(ii) P is a finite set of production rules of the form $A \to B$ where $A, B \in (\Sigma \cup N)^*$ and A contains at least one nonterminal symbol.

(iii) S is a starting symbol and $S \in N$.

The language $L(G)$ generated by G is defined by $L(G) = \{A : A \in \Sigma^* \text{ and } S \overset{*}{\Rightarrow} A\}$.

A grammar $G = (\Sigma, N, P, S)$ is called a *context-free grammar* (CFG) if each production rule in P is of the form $A \to \alpha$ where $A \in N$ and $\alpha \in (\Sigma \cup N)^*$. The language generated by a CFG is called a *context-free language* (CFL).

Several applications such as natural language processing and recognition of natural unrestricted speech requires a language

model that attaches probabilities to the production rules. This naturally leads to the concept of probabilistic context-free grammar, which is used as a model for research in security and privacy.

A probabilistic context-free grammar (PCFG) is a context-free grammar $G = (\Sigma, N, P, S)$ in which for each production rule $A \to \beta$ where $A \in N$ and $\beta \in (\Sigma \cup N)^*$, a probability $q(A \to \beta)$ is assigned satisfying the condition

$$\sum_{(A \to \beta) \in P} q(A \to \beta) = 1 \text{ for all } A \in N.$$

A conjunctive grammar is a quadruple $G = (\Sigma, N, P, S)$, in which Σ and N are disjoint finite non-empty sets of terminal and nonteriminal symbols respectively, P is a finite set of rules of the form

$$A \to \alpha_1 \& \cdots \& \alpha_n (\text{with } A \in N \text{ and } \alpha_1, \cdots, \alpha_n \in (\Sigma \cup N)^*) \qquad (*)$$

and $S \in N$ is a nonterminal designed as the start symbol. For any rule of the form $(*)$ and for any number $i (1 \leq i \leq n)$, an object $A \to \alpha_i$ is referred to as a conjunct.

We now present some basic concepts in Tile Latching System.

A tile is a two-dimensional polygon in which each side is of unit length. In a tiling pattern a side of one tile is latched with a side of another tile. In a tiling pattern generated by using n tiles, two tiles are adjacent if they have a common edge.

A latching system is a triple $S = (\Sigma, P, t_0)$ where Σ is a finite nonempty set of regular polygonal tiles in which at least one side of each tile is latchable, P is a finite set of latching rules and t_0 is the starting symbol.

A pattern p_2 is generated from a pattern p_1 by applying the latching rules in parallel to any of the latchable sides on the boundary of p_1. The set of all patterns generated from the starting tile constitutes the pattern language $L(S)$ of S. A tiling pattern can be considered as a planar graph whose vertex set and edge set are the vertices and sides of the polygonal tiles of the pattern. Hence graph theoretical tools and terminologies can be used to describe the language $L(S)$.

In Chapter 2, we investigate a triangular tile latching system. Let Σ be a set of equiangular triangular tiles with at least one latchable side. The attachment rule P_1 permits two tiles to get latched along a latchable side. In this chapter we determine the language generated by a triangular latching system using the concept of geometric dual of a planar graph.

Let $A = \{0, 1\}$ be a binary set of symbols, where 1 represents the possibility of latching and 0 otherwise. Let T be a triangular tile whose sides are labelled from A.

A labelled tile T is said to be i-side latchable if i sides are labelled with 1, where $1 \leq i \leq 3$.

The graph $G = K_4 - e$ is called a dimer. An attachment rule P_1 is a rule which attaches the edges with label 1 of tiles T_1 and T_2 to form a dimer in which the latched edge is unlabeled. This implies that no further latching is possible with that side.

The attachment rule P_1 is given in Figure 1.

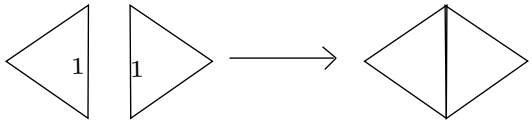

Figure 1: Attachment rule giving a dimer

A latching system is a triple S= (Σ, P_1, t_0), where Σ is a set of i-side latchable triangular tiles where $1 \leq i \leq 3$, the tile t_0 in Σ is the starting symbol and P_1 is the attachment rule given in the above Definition.

Theorem 2. *Let $S = (\Sigma, P_1, t_0)$ be a latching system where Σ is a set of 1-side latchable tiles and $|\Sigma|$=k. Then $G \in L(S)$ if and only if G is a plane graph with $\lceil k/2 \rceil$ components, at most one component is a triangle and each of the remaining components is a dimer.*

Let G be a planar graph and let G^* be the geometric dual of G. Let $H^* = G^* - v_0$, where v_0 is the vertex in the exterior face of G.

A polygonal tiling pattern G is called a linear polygonal tiling pattern if H^* is a path.

Theorem 3. *Let $S = (\Sigma, P_1, t_0)$ be a latching system where Σ is a set of 2-side latchable tiles and $|\Sigma|$=k. Then $G \in L(S)$ if and only if G is a linear polygonal tiling pattern.*

Theorem 4. *Let $S = (\Sigma, P_1, t_0)$ be a latching system where Σ is a set of 3-side latchable tiles and $|\Sigma|$=k. Then $G \in L(S)$ if and only if G is a connected plane graph with $k+2$ vertices, $2k+1$ edges and $k+1$ faces and all faces except the exterior face are triangles.*

Our investigation shows that a connected plane graph with an annular region cannot be generated by the tile latching system considered in this chapter 2. The following problems arise naturally.

Problem 1. *Let $S = (\Sigma, P_1, t_0)$ where $\Sigma = \Sigma_1 \cup \Sigma_2 \cup \Sigma_3$ where Σ_i is a set of i-side latchable sides where $1 \leq i \leq 3$. Determine the language $L(S)$ generated by S.*

For the triangular tile latching system S considered in Chapter 2, any single annular tiling pattern does not belong to $L(S)$. To address this problem in Chapter 3 we introduce two more productions rules, namely, the insertion rule and touching rule and prove that in this latching system a single annular tiling pattern can also be generated.

An insertion rule P_2 is a rule in which three i-side latchable tiles are used to form a tiling pattern as shown in Figure 2.

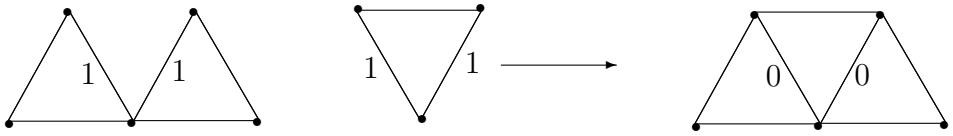

Figure 2: The insertion rule P_2

A touching rule P_3 is a rule in which four latchable tiles T_1, T_2, T_3 and T_4 are used to form a tiling pattern G_4 as follows. Let D_1 be the dimer obtained from T_1 and T_2 by applying the attachment rule P_1. Let D_2 be the dimer obtained from T_3 and T_4 by applying the attachment rule P_1. Let v_1 and v_2 be vertices lying on a latchable side of D_1 and D_2 respectively. Then the tiling pattern G_4 is obtained from D_1 and D_2 such that the vertex v_1 touches the vertex v_2.

The formation of the tiling pattern G_4 is given in Figure 3

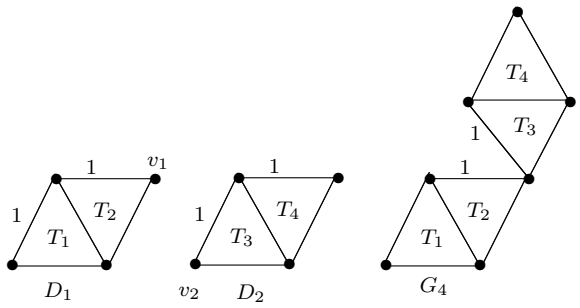

Figure 3: The pattern G_4 obtained by the touching rule

The concept of touching can also be thought of as a vertex v_1 getting latched with a vertex v_2.

Let $S_1 = (\Sigma, \{P_1, P_2, P_3\}, t_0)$ where P_1 is the attachment rule, P_2 is the insertion rule and P_3 is the touching rule.

Theorem 5. *Let* $S_1 = \{\Sigma, \{P_1, P_2, P_3\}, t_0\}$ *be a tile latching sys-*

tem where Σ is a set of 2-side latchable tiles and $|\Sigma| = n$. Let G_1 be a tiling pattern obtained from r tiles $T_1, T_2, \ldots, T_r$ by applying successively the attachment rule P_1.

(i) The tiling pattern G_1 has exactly two latchable sides $e_1 = u_1v_1$ and $e_2 = u_2v_2$ where e_1 lies on T_1 and e_2 lies on T_r.

(ii) The touching rule P_3 can be applied to G_1 if and only if one of the vertices u_1 and v_1, say u_1 is such that $d(u_1, u_2) = d(u_1, v_2) = 1$. Let G_2 be the tiling pattern obtained from G_1 by applying P_3 using the tile T_{r+1}. Then the tiling pattern G_2 satisfies the following properties.

(a) G_2 has exactly two latchable sides.

(b) G_2 is a single annular tiling pattern.

(c) The insertion rule P_2 can be applied to G_2 giving a tiliting pattern G_3 which is also a single annular tiling pattern.

(d) If k is the length of the cycle forming the boundary of the annular region of G_2, then $k - 1$ is the length of the cycle forming the boundary of the annular region of G_3.

(iii) The insertion rule P_2 can be applied to G_1 if and only if e_1 and e_2 are adjacent edges and the angle between them is $\frac{\pi}{3}$. The resulting tiling pattern G_4 has no latchable side and G_4 is either a solid hexagon or a single annular tiling pattern.

Corollary 1. *Let $L(S_1)$ be the language generated by the tile latching system S_1. Then $G \in L(S_1)$ if and only if G is a linear polygonal tiling pattern or a solid hexagon or a single annular tiling pattern.*

In Theorem 5 and Corollary 1, Σ is a set of 2-side latchable tiles. Hence the following problem arises naturally.

Problem 2. *Let Σ be a set of 3-side latchable. Let $S_2 = (\Sigma, \{P_1, P_2, P_3\}, t_0)$. Determine $L(S_2)$.*

We observe that a multi annular tiling pattern can also be generated by S_2.

In several applications such as natural language processing and recognition of natural unrestricted speech, it becomes necessary to attach probabilistic to the production rules. This resulted in the concept of probabilistic context-free grammar (PCFG).

Motivated by the concept of probabilistic context-free grammar, in Chapter 4 we propose a probabilistic model of conjunctive grammar which can be used in computational linguistics, security and privacy in computer science. This model will be more effective than PCFG in the above areas of application, since conjunctive grammar is a powerful extension of context-free grammar.

A Conjunctive grammar $G = (\Sigma, N, P, S, q)$, is called a probabilistic conjunctive grammar (PCG) if for each rule $A \to B$, a probability $q(A \to B)$ is assigned such that for any $A \in N$,

$$\sum_{A \to \beta \in P} q(A \to \beta) = 1.$$

Also if T is a parse tree containing the rules $A_1 \to \beta_1,\ A_2 \to \beta_2, \cdots,$ $A_n \to \beta_n$ then

$$p(T) = \prod_{i=1}^{n} q(A_i \to \beta_i).$$

Assigning probabilities is important when dealing with machine learning because a string can have many parse trees to derive it. These parse trees will possibly have very different structures but the application of probabilities helps us to select the most probable parse tree and the parsing will be faster by pruning off the low probability subtrees.

The probabilistic conjunctive language $L(G_p)$ defined by a PCG is given as
$L(G_p) = \{(x, q(x)) \mid S \xrightarrow{q_i(x)} x, \quad for\ i = 1, \cdots, k, \quad x \in \Sigma^* \ and$ $q(x) = \sum_{i=1}^{k} q_i(x)\}$
where S is the start symbol and there are k distinctively different derivations of generating x from S.

A PCG is said to be in binary normal form, if each rule in P is one of the following forms:

(i) $A \to B_1C_1\& \cdots \&B_mC_m$, where $m \geq 1; A, B_i, C_i \in N$,

(ii) $A \to a$, where $A \in N, a \in \Sigma$,

(iii) $S \to \epsilon$, if S does not appear in right parts of the production rules.

Each rule r in P is augmented with a probability $q(r)$ such that for any $A \in N$, $\sum_{r \in P} q(r) = 1$.

Theorem 6. *Any PCG G_p is equivalent to a PCG G_{p_b} in which all the productions are in binary normal form.*

Let $G = (\Sigma, N, P, S)$ be a context-free grammar. The problem of assigning probability q to the production rules in such a way that (Σ, N, P, S, q) is a probabilistic context-free grammar is a subtle task. Chi and Geman [5] investigated this problem and proved that for a context-free grammar, the maximum likelihood estimator of production probabilities for a sample data impose probabilities on the production rules in such a way that the resulting system is a probabilistic context-free grammar. In Chapter 5 we adopt a similar approach for estimating and assigning probabilities to the production rules of a conjunctive grammar so as to make it a probabilistic conjunctive grammar.

In Chapter 6 a password cracking algorithm is designed using Probabilistic Conjunctive Grammar (PCG). Here PCG is used for deriving the base structure of target password by combining multiple parse trees. After analyzing the base structure of each password in the data-set, the base structures which are similar to the base structures of the target passwords are collected. In the final step the password is cracked with its cracking probability.

The running time of the algorithm presented in this chapter is $O(n^2)$. This algorithm can be applied in various fields associ-

ated with security and privacy. Our further research direction is to explore password strength using this approach.

In Chapter 7, which is the concluding chapter of this thesis, we indicate open problems and directions for further research which arise naturally from our investigations.

References

[1] G. Chartrand, L. Lesniak and P. Zhang, Graphs & Digraphs, Sixth edition, CRC Press, London, (2016).

[2] J.E. Hopcroft, R. Motwani and J.D. Ullman, J. Introduction to automata theory, languages, and computation, *Acm Sigact News.*, 32(1) (2001), 60–65.

[3] A. Salomaa, *Formal languages*, Academic Press Professional, Inc. (1987).

[4] A. Meduna, *Automata and languages: theory and applications*, Springer Science & Business Media, (2012).

[5] Z. Chi and S. Geman, Estimation of probabilistic context-free grammars, *Computation linguistics* **24(2)** (1998), 299–305.

LIST OF PUBLICATIONS

1. K. Kanchan Devi and S. Arumugam, Assignment of Probabilities to the Production Rules of A Conjunctive Grammar, *Soft Computing*,(2023) DoI: 10.1007/s00500-023-08707-7 (SCI with IF: 3.732)

2. K. Kanchan Devi and S. Arumugam, Triangular Tile Latching System, *Communications in Combinatorics and Optimization*, **9**(2) (2024), 309-316. (Scopus)

3. K. Kanchan Devi and S. Arumugam, *Password Cracking Algorithm using Probabilistic Conjunctive Grammar*, IEEE International Conference on Intelligent Techniques in Control, Optimization and Signal Processing, INCOS 2019, (2019), art. no. 8951390. (Scopus)

4. K. Kanchan Devi and S. Arumugam, Probabilistic conjunctive grammar, *Lecture Notes in Computer Science*, *10398* (2017), 119–127.(Scopus)